Lecture7. Deadlock

1> deadlock: when two/more processes are waiting indefinitely for an event that can be caused only by one of the waiting processes

2> 4 conditions : Mutual Exclusion, Hold and Wait, No preemption, Circular Wait

3> Three principle methods for handling deadlock.

3-1> 데드락으로 못들어가게 하기 : prevention, avoidance

3-2> 데드락 들어가면 recover : detection, recovery

3-3> 데드락 안일어난척하기

3-1>

2) Deadlock avoidance : requires a priori information, such as the maximum number of each resources classes.

: Resource-Allocation Graph algorithm, Bankers algorithm

3-2>

1)When deadlock is detected, the system must recover either

by termination some of the deadlocked process (데드락 걸린 프로세스 종료)

or by preempting resources from some of the deadlocked processes (데드락 걸린 프로세스의 자원 선점해버림)

1> deadlock

- when two/more processes are waiting indefinitely for an event that can be caused only by one of the waiting processes

- a set of blocked process each holding a resource/ waiting to acquire a resource/ held by another process in the set.

ex) 세마포어 변수 A,B 둘다 1로 초기화중이면

```
$P_0$            $P_1$
wait (A);        wait(B)
wait (B);        wait(A)
```

* starvation is possible?

2> Deadlock Characterization

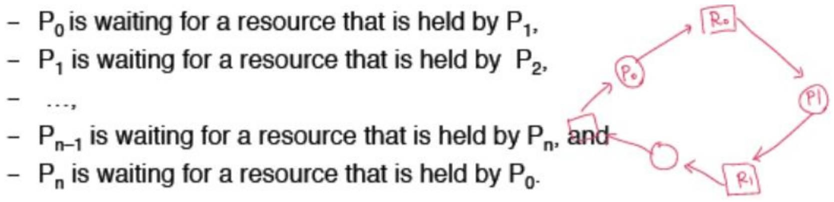
: Deadlock can arise if four conditions hold simultaneously (not the condition for deadlock)

1. Mutual Exclusion: Only one process at a time can use a resource

2. Hold and Wait: a process holding at least on resource is waiting to acquire additional resources held by other process

3. No preemption: a resource can be released only voluntarily by the process holding it, after that process has completed its task

4. Circular Wait:



- P_0 is waiting for a resource that is held by P_1,
- P_1 is waiting for a resource that is held by P_2,
- …,
- P_{n-1} is waiting for a resource that is held by P_n, and
- P_n is waiting for a resource that is held by P_0.

3> Methods for Handling Deadlocks

1. 데드락으로 못들어가게 하기 : prevention, avoidance

2. 데드락 들어가면 recover : detection, recovery

3. 데드락 안일어난척하기

데드락으로 못들어가게 하기: prevention, avoidance

3-1> 데드락으로 못들어가게 하기 : prevention, avoidan

1) Deadlock Prevention: 4조건중 하나를 막아서 데드락 예방하기

2) Deadlock avoidance (=DA)

: requires a priori information, such as the maximum number of each resources classes.

: Resource-Allocation Graph algorithm, Bankers algorithm

- Deadlock Avoidance requires the system provides some additional information in advance about how resources to be requested

- Simplest: each process declare the maximum number of resources of each type that it may need

- The deadlock-avoidance algorithm dynamically examines the resources-allocation state

  - circular-wait condition 절대 안되게 하려고

- Resource-allocation state is defined by the number of available and allocated resources, and the maximum demands of the processes.

<Resource Allocation State: Safe, Unsafe, Deadlock>

deadlock(circular-wait & deadlock) ⊂ unsafe (circular-wait but no deadlock) | safe(no circular-wait)

* unsafe state면 데드락 가능성이 있는거고 이 unsafe에 안들어가게 하는게 Avoidance

<Safe State>

- When a process requests an available resource,

  - system must decide if immediate allocation leaves the system in a safe state.

- 시스템이 safe state에 있다는 것은- safe sequence of all processes